A Kernel Model with Conditional Moving Windows for the Prediction of Transmembrane Helices in Proteins

Massimo Mucciardi[1], Cinzia Di Salvo[2], Giovanni Pirrotta[3]

[1]Department of Economics, Statistics, Mathematics e Sociology, University of Messina, Italy

[2]Department Animal Biology and Marine Ecology, University of Messina, Italy

[3]Department of Social and Human Sciences, University of Messina, Italy

Emails: [1]massimo.mucciardi@unime.it; [2]cdisalvo@unime.it; [3]gpirrotta@unime.it

Abstract

The determination of some proteins structure at high resolution often results difficult from an experimental point of view. It is true especially for many integral membrane proteins and intrinsically disordered proteins. In this context, we try to develop a statistical technique alternative to the classical methods used to calculate hydropathy graphics, called Kernel Windows Method with Conditional Moving Windows (KWMCMW); the latter has been implemented in order to improve the degree of accuracy of the transmembrane helices (TMH) prediction. The KWMCMW introduces the effect of distance in calculating the hydropathy value for each amino acid (AA) residue in the sequence. Consequently, this new method is extended to a peripheral membrane protein with partially unknown structure. The implementation of specific software written in Python language makes the method suitable for application to any membrane protein.

Keywords

Kernel Function; Transmembrane Helices Prediction; Hydropathy Scale; Conditional Moving Windows; Python Language

Introduction

Computational algorithms in conjunction with statistical analysis are useful methods to identify transmembrane helices (TMH) in a protein, since many of these remain recalcitrant to structure determination by X-ray crystallography. So different methods have been developed in order to recognize regions of the proteins that can be ascribed to α-helix, β-strands or random coil [1-2]. Generally, secondary structure prediction algorithms can be categorized into three main classes: physicochemical methods based on various hydropathy scales, statistical methods, and machine learning methods based on various learning algorithms such as Neural Networks, Hidden Markov Models, and Support Vector Machines. As regard the physicochemical methods, one of the first approaches is provided by Kyte and Doolittle (KD) [3]. This method, through the study of the graphics of hydropathy, gives a suitable accuracy in the determination of TM helices, and is based on a scale where the hydrophilic and hydrophobic properties of each of the 20 AA side-chains are taken into consideration. The method uses a moving (or sliding) window (MW) that continuously determines the average of the hydrophaty within a segment of predetermined length L as it advances through the sequence. Recently different prediction programs for TM segments have been developed. Among these, Yizhou Li et. al. [4] in order to identify the signal peptide of a protein or predict its cleavage site, implement an automated method by using artificial neural network. In doing this, hydrophaty scale of KD is adopted to code each AA. In fact is reported that the cleavage site has relationship with the neighboring sequence environment, i.e., hydrophobic core h-region.

Based on the selected physicochemical properties, in this paper we show an advance of the Kernel Windows Method (KWM) [5] in order to improve the degree of accuracy of the prediction of TM helices. The method combines the chemical properties and statistical methodology so as to keep into account the proximity between the AA residues along protein sequence. Consequently, we made an application of the method on Myelin Basic Protein (MBP) which belongs to the family of intrinsically disordered proteins, which fail to form rigid 3-D structures under physiological conditions. In fact, although several structural and computational studies [6-7-8] have elucidated on the secondary structure of the protein, no evidences on the tertiary structure have yet been obtained. Moreover, intrinsically disordered proteins with nonpolar residues can also penetrate the interfacial region of the membrane and reach the hydrophobic core, especially when such proteins, as MBP, are cationic and interact with negatively charged membrane [9]. For these reasons, we use the KWM in order to show if the method could be applied as identifier of the secondary structure element, in particular α-helices interacting with membrane, in MBP. The paper is organized as follows. Section 2 presents the KWM and the advances introduced, section 3 shows the main results obtained using the KWM considering the database of 92 proteins used and the results of the prediction on MBP, finally in the appendix we provide the code used to implement the KWM algorithm.

The Kernel Windows Method (Kwm): Methodology and Advances

Before introducing the solution proposed for the prediction of secondary structure in TM, we should briefly summarize the KWM method [5]. The term 'kernel' refers to non-parametric methods that involve calculations using a well-defined local neighbourhood. So, the kernel is utilized as smoothing filter that keeps in count the distance between AA residues along protein sequence: we assume that the values of hydropathy in the neighbourhood of the central residue are more influential than distant ones. Therefore, the KWM is different to the MW method where all residues have the same weight (1/L) with respect to the central residue. From this point of view, KWM can be seen as a special case of MW. Other features are: a) KWM uses a single sequence as input; b) KWM is a non-parametric model since it does not assume a specific pattern for AA residues in the membrane. So under this assumption, the weights are chosen in order to assign greater importance to nearest residues by using a univariate symmetric kernel function. Let 'L' be the length of the MW and 'i' the position of the residue; then we define the value of kernel hydropathy (Δi) for the i-th residue as:

$$\Delta_i = \sum_{j=-\left(\frac{L-1}{2}\right)}^{\left(\frac{L-1}{2}\right)} H_j K_j \tag{1}$$

For the j-th residue within the window L, H_j is the value of hydrophaty and K_j is the value of the kernel function. Under this aspect we can observe that KWM works as a two-stage system. First the length (L) of the window is considered and then the kernel function is applied.

The value K_j is determined according to a distance decay model as follows:

$$K_j = \frac{f\left(\left|j-i\right|\right)}{A} \tag{2}$$

In this case the distance $\left|j-i\right|$ in **Eq.2** is calculated as the difference (in absolute value) between the position of the residue 'j' and the position of the central residue of reference 'i' within the width of the MW chosen. Currently the functions $f\left(\left|j-i\right|\right)$ considered and implemented are:

1) Gaussian distance decay function;

$$f\left(\left|j-i\right|\right) = e^{-\varphi\left(\frac{\left|j-i\right|}{h}\right)^{\alpha}} \tag{3}$$

2) Bi-square distance decay function;

$$f\left(\left|j-i\right|\right) = \left(1-\left(\frac{\left|j-i\right|}{h}\right)^2\right)^2 \tag{4}$$

and Epanechnikov function;

$$f\left(\left|j-i\right|\right) = \frac{3}{4}\left(1-\left(\frac{\left|j-i\right|}{h}\right)^2\right) \tag{5}$$

In Gaussian function ϕ is the decay parameter, α is a smoothing parameter; usually these parameters are set with ϕ =0.5 and α =2. For all three models h is a non negative parameter, called bandwidth, that controls the width of the kernel [10]. Therefore it is reasonable to assume h<L. A small bandwidth will produce a sharp kernel with many variations in the weights, while a high bandwidth will produce a flat kernel with little variation in the weights. In this last case it is easy to verify that the KWM tends to MW. Note that for the above models when i=j (central residue), the numerator of K_j assumes the value equal to 1 for Gauss and Bi-square functions and 0.75 for Epanecnikov function. Moreover, for all functions, in

order to ensure that the sum of all weights is equal to 1, we divide by the constant A, where A is equal to:

$$A = \sum_{j=-\left(\frac{L-1}{2}\right)}^{\left(\frac{L-1}{2}\right)} f\left(\left|j-i\right|\right) \tag{6}$$

and so for the i-th residue

$$\sum_{j=-\left(\frac{L-1}{2}\right)}^{\left(\frac{L-1}{2}\right)} K_j = 1 \tag{7}$$

.

From this point of view, this kernel is also called 'order 0 kernel' [11]. In addition, for the first and the last residues of the sequence the kernel is automatically calculated with smaller windows. Despite the KWM method can work with any value of the window size (L), in this work we use a model with a conditional windows (KWMCMW) that vary between 11 AA and 27 AA of length to make it compatible with the number of residues required to span the membrane in a helical conformation [1]. Under this condition, the algorithm implemented allows to find the best values of L, h and threshold (Th_I) that optimize a special parameter of 'smooth'[1]. Next, other two thresholds are calculated in order to maximize the values of F-measure and MCC [12]. We call these threshold values Th_F and Th_M respectively (see appendix for more details).

Principal Results

In order to compare the performance of KWMCMW method with a classical MW method (reference model), the same training dataset proteins has been used. We select 92 non redundant protein sequences with experimentally determined transmembrane topology from 3D_helix MPtopo database [13]. Although we used various types of kernel, in this research we present only the results obtained with the Gaussian KWMCMW setting ϕ=0.5 and α=2 (table 1). For the MW model (table 2) we use the same software of the KWMCMW model but with a fixed window equal to 19 AA and a fixed hydropathy threshold greater than 1.6 [1]. For the hydropathy scale of each residue we use the one provided by Kyte and Doolittle [3]. The performance of classification is assessed by these indices: accuracy (ACC), precision (PR), sensitivity (SN, also called the recall), specificity (SP), the correlation coefficient of Matthew (MCC) and F-measure (F-m) [12]. These indices are calculated using the number of true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN) for each classifier. In addition other two indices have been calculated: Coverage (CVR), the ratio between the number of predicted residues and known residues inside the helices; Helix rate (H_r), the ratio between the number of predicted helices and known helices. As can be seen from the comparison of the two tables (table 1 and 2), the KWMCMW method gives better results in all performance indices. Note that this new method provides a low number of false positive compared to the classic method MW.

As mentioned above, in KWMCMW the influence between residues is measured by weighting system using the functions above mentioned. So, to verify that KWMCMW may be applied for the prediction of secondary elements such as α-helices in protein MBP (Human Isoform 5 identifier code P02686-5), we select from the results obtained for the model KWMCMW (table 1) only those proteins (N=34) compatible with the length of the protein MBP (Lp = 170 AA). In this way we obtain a value of L = 15.91, h = 3.59, Th_I=1.10, Th_F = 0.03 and Th_M=0.59. The application of these parameters to KWMCMW (with Th_F=0.03) provides the results shown in fig 1. Consequently, when MW method is applied to the protein, no peaks above the threshold appear (fig. 1). On the contrary, as can be seen in fig. 1, the shape of the KWMCMW evidences the presence of different peaks. As mentioned above, no X ray resolution is yet available for MBP, but divergent results have been obtained from prediction studies or structural ones. In our analysis the KWMCMW method used agrees with recent literature in the prediction of three peaks in particular: the first peak revealed at residues 38-43 is in accordance with the presence of an anphyphatix α-helix, in which the two sets of phe-phe pairs at the position 42-43 are at least partially buried and immobilized in the membrane as revealed by solid–state NMR spectroscopy studies [14-15].

[1] In the first experimental version of the KWM [5] the values of L and h, were obtained by minimizing the coefficient of variation (CV) of the hydrophobicity inside the α-helices while the threshold was fixed at 0.

TABLE 1 STATISTICS FOR KWMCMW (92 PROTEINS)

Statistics	L	h	Th_I	Max_F	Th_F	Max_	Th_	ACC	PR	SN	SP	CVR	H_r
N	92	92	92	92	92	92	92	92	92	92	92	92	92
Mean	17.74	3.97	1.37	0.84	0.22	0.69	0.71	77.14	99.13	52.67	99.83	53.31	0.99
Median	17.00	3.80	1.36	0.85	0.38	0.69	0.76	79.34	100.00	50.61	100.00	52.47	1.00
Mode	11.00	2.90	-0.34	1.00	-1.34	1.00	0.80	100.00	100.00	100.00	100.00	100.00	1.00
Std. Dev.	5.384	1.67	0.62	0.10	0.81	0.17	0.63	17.13	5.38	23.48	0.78	23.74	0.37
Min	11.00	1.30	-0.34	0.46	-2.29	0.24	-0.92	16.46	48.84	4.35	93.99	4.35	0.67
Max	27.00	12.20	2.81	1.00	1.88	1.00	2.77	100.00	100.00	100.00	100.00	100.00	4.50
25° Perc.	13.00	2.90	1.03	0.79	-0.26	0.44	0.36	64.81	100.00	38.06	100.00	38.34	1.00
75° Perc	23.00	4.90	1.73	0.90	0.77	0.75	1.13	91.07	100.00	67.03	100.00	68.22	1.00

Total know helices =310 Total predicted helices =326 Max_F = Max F-Measure Max_M = Max MCC

TABLE 2 STATISTICS FOR MW (92 PROTEINS)

Statistics	L	h	Fm	Th	MCC	Th_M	ACC	PR	SN	SP	CVR	H_r
N	92	92	92	92	92	92	92	92	92	92	92	92
Mean	19.00	-	0.42	1.60	0.34	0.50	65.60	88.71	30.33	95.91	33.11	1.54
Median	19.00	-	0.43	1.60	0.32	0.57	65.33	95.96	27.43	99.63	28.82	1.41
Mode	19.00	-	0.80	1.60	0.00	0.55	58.14	100.00	0.00	100.00	66.67	2.00
Std. Dev.	.00	-	0.22	.00	0.19	0.59	17.24	19.40	20.48	9.62	22.23	0.85
Min	19.00	-	0.00	1.60	0.00	-2.17	26.14	-1.00	0.00	42.86	0.00	0.00
Max	19.00	-	0.93	1.60	0.80	1.53	94.57	100.00	100.00	100.00	116.00	5.00
25° Perc.	19.00	-	0.27	1.60	0.21	0.27	51.18	87.95	15.97	96.01	18.24	1.00
75° Perc	19.00	-	0.59	1.60	0.49	0.86	81.31	100.00	44.16	100.00	46.34	2.00

Total know helices =310 Total predicted helices =421

The peak centred at residues 85-93 corresponds to an α-helix immersed in the bilayer identified by studies of electronic paramagnetic resonance (EPR) [16]. The third peak (148-155) corresponds to the α-helix Calmodulin binding domain [15]. However, apart from these results, we obtain a better definition of the shape respect to the MW method.

Conclusions

From these results we can assert that the KWMCMW, giving a different weight on the hydropathy of residues according to the distance-decay function, well identifies helices in membrane proteins, so it can be considered an alternative method to other existing bioinformatics and statistical methods. Moreover, it also gives useful information on some secondary elements identified as α-helices of MBP, which interact with oligodendrocyte membrane. So, our prediction is in agreement with the presence of hydrophobic α-helices in MBP, as revealed by the recent structural analyses. Further studies could try to apply KWMCMW on others intrinsically disordered proteins with the aim to identify these secondary elements as well. From a statistical point of view, it is evident that it may be necessary to improve the current criterion for estimating the threshold and to implement new kernel functions combining them with other scales of hydropathy. However, further progress will require more learning set sequences. Some researches of this kind are currently in progress by authors.

Appendix: Implementation of the Kwmcmw Algorithm

The KWMCMW algorithm target is to find the best protein threshold able to locate all the segment helices in the protein implementing the kernel function of the **Eq.1**. The algorithm, implemented in Python 2.7 [17] using Scipy and Numpy external scientific libraries, works as follows.

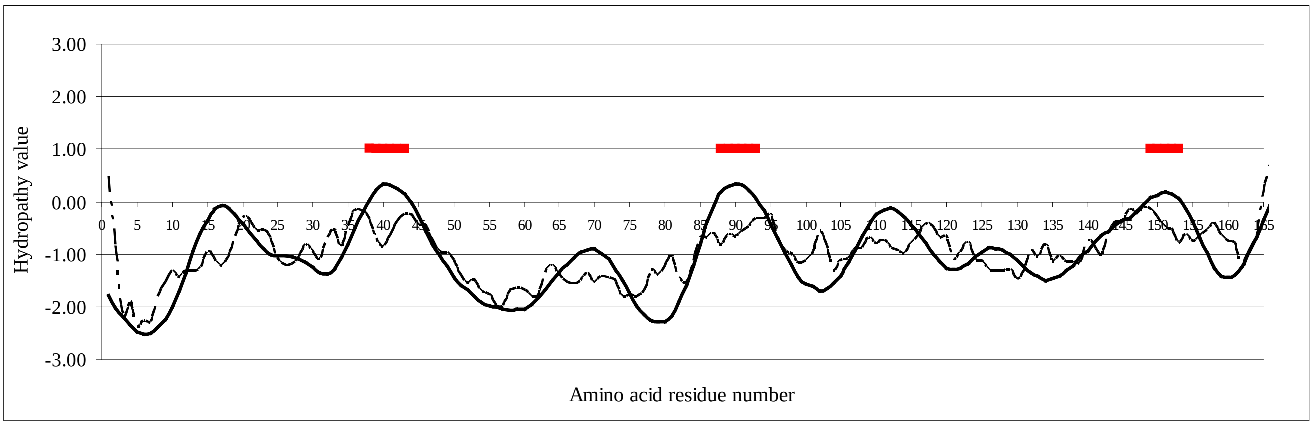


FIG. 1 TM HELICES PREDICTION PROFILES OF HYDROPHOBICITY FOR MBP: SOLID LINE GAUSSIAN KWMCMW; DASHED LINE CLASSICAL MW.THE THREE RED LINES AT THE TOP INDICATE THE POSITIONS OF PREDICTED A-HELICES WITH THE GAUSSIAN KWMCMW

Primary, the algorithm loads the sequences of proteins and the relative transmembrane topology. Second, the algorithm defines the kernel type and the hydrophobic scale: Hydrophobicity, Helicity, KD, GES, and Eisenberg [18]. Only for Gaussian kernel, the alpha and phi values are calculated. Third the algorithm generates all possible combinations of L size (between 11 AA and 27 AA) and h (0<h<L) producing kernel values of hydropathy. Knowing a-priori the transmembrane topology, the algorithm calculates, for each helix (H) and no-helix (NH) segment, some statistics (Mean, Median, Mode, SD, Max, Min and Quartile). So, we consider the following initial threshold: Th*=Max[Max(NH)]. At this point, in order to ensure a very high percentage of precision in the segment classification, the following condition (filter) is adopted: Th*<Min[Max(H)]. This condition creates a restriction in the number of initial combinations of L and h and is essential because it eliminates all those values of hydrophobicity present in the no-helix segments that exceed the values of hydrophobicity in the helix segments. Only for the values of L and h that produce hydrophobic kernel values that exceed the condition, the algorithm computes a special parameter called 'irregularity' to estimate the degree of 'smooth' of kernel. As can be observed in the routine below, this parameter is initially set to a value intentionally large (Irr = 1000). Cycling for all L and h values, the algorithm finds the better threshold (Th_I) that minimizes the 'irregularity' property. In addition, starting from the lower kernel value up to threshold found (Th_I), the algorithm calculates further two new threshold values considering and maximizing MCC and F measures. Therefore, the final output of the algorithm returns 3 threshold values (Th_I, Th_F and Th_M) which can be used to make predictions (see fig. 2 and 3). Currently the library is in an alpha version while the beta version will be released as soon as possible.

```
FOUND = FALSE
MIN_IRR = 1000
LOAD Protein
FOR i=11 TO 27 STEP 2 DO:
  FOR j=1 TO i STEP 0.1 DO:
generator=GaussianKernelGenerator(proteine, scale, i, j, alpha, phi)
    stat = STATISTICS(generator)
    IF stat [MAX-MAX-NH]<stat [MIN-MAX-H] THEN
      Th* = stat[MAX-MAX-NH]
      FOUND = TRUE
      IRR = GetIrregularity(Th*)
      IF IRR < MIN_IRR THEN
        MIN_IRR = IRR
        L = i
        h = j
        Th_I = Th*
        IF MIN_IRR == 0 THEN
          BREAK
IF FOUND THEN
  NORMALIZE(Th_I)
  Th_M = MAX-MCC-MEASURE(Th_I)
  Th_F = MAX-F-MEASURE(Th_I)
```

FIG. 2 CODE OF THE KWMCMW ALGORITHM

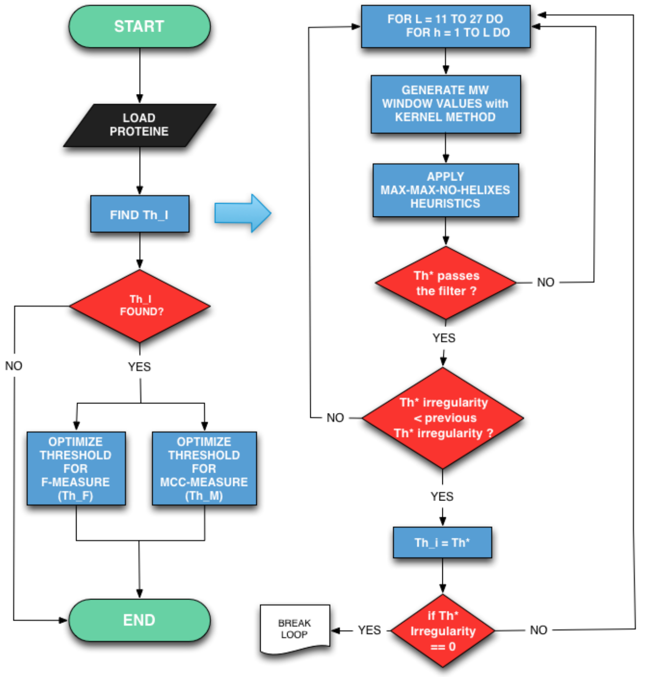

START
LOAD PROTEINE
FIND Th_I
Th_I FOUND?
NO
YES
OPTIMIZE THRESHOLD FOR F-MEASURE (Th_F)
OPTIMIZE THRESHOLD FOR MCC-MEASURE (Th_M)
END
FOR L = 11 TO 27 DO
FOR h = 1 TO L DO
GENERATE MW WINDOW VALUES with KERNEL METHOD
APPLY MAX-MAX-NO-HELIXES HEURISTICS
Th* passes the filter ?
NO
YES
Th* irregularity < previous Th* irregularity ?
NO
YES
Th_i = Th*
if Th* Irregularity == 0
YES
NO
BREAK LOOP

FIG. 3 FLOW CHART OF THE KWMCMW ALGORITHM

REFERENCES

[1] X. Hu, 'Structure Prediction of Membrane Proteins', in Computational Methods for Protein Structure Prediction and Modeling, Editors Y. Xu, D. Xu, H.Liang, pp 65-108, Vol 2, Springer, (2007).

[2] C. P. Chen, A. Kernytsky, and B. Rost, 'Transmembrane helix predictions revisited', Protein Science, 11, (2002).

[3] J. Kyte, R.F. Doolittle, 'A Simple Method for Displaying the Hydropathic Character of a Protein', J. Mol. Biol., vol .157 (1982).

[4] L. Yizhou , W. Zhining, Z. Cuisong, T. Fuyuan, L. Menglong, 'Effects of neighboring sequence environment in predicting cleavage sites of signal peptides', Peptides, vol 29, pp. 1498 – 1504, (2008).

[5] M. Mucciardi, C. Di Salvo, 'A kernel windows method for prediction of protein hydropathy. An application on a myelin associated protein', International Conference on Management Sciences and Information Technology (MSIT 2012), June 1-2, Changsha, China.

[6] R. A. Ridsdale, D. R. Beniac, T. A. Tompkins, M. A. Moscarello, G. Harauz, 'Three-dimensional Structure of Myelin Basic Protein', Journal of Biological Chemistry Vol. 272, No. 7, pp. 4269–4275, (1997).

[7] G. L. Stoner, 'Predicted folding of beta-structure in myelin basic protein', J Neurochem, Aug; 43(2), pp.433-47, (1984).

[8] R. Martenson, 'Prediction of the secondary structure of myelin basic protein, J. Neurochem', Apr 36 (4), pp1543-60, 1981.

[9] J.F. Ellena, J. Moulthrop, J. Wu, M. Rauch, S. Jaysinghne, J.D. Castle, D.S. Cafiso, 'Membrane position of a basic aromatic peptide that sequesters phosphatidylinositol 4,5 bisphosphate determined by site-directed spin labeling and high-resolution NMR'. Biophys J. 87 (5): 3221–3233, (2007).

[10] B.W. Silvermann, 'Density Estimation for Statistics and Data' Analysis, Chapman and Hall, New York, (1986).

[11] A.J. Izenman, 'Recent Developments in Nonparametric Density Estimation', Journal of the American Statistical Association , Vol. 86, No. 413, pp. 205-224, (1991).

[12] Y. Xiong, J. Liu, W. Zhang, T. Zeng, 'Prediction of heme binding residues from protein sequences with integrative sequence profiles', IEEE International Conference on Bioinformatics and Biomedicine, Proteome Science, (2012).

[13] http://blanco.biomol.uci.edu/, S. White Laboratory, (2011).

[14] L. Zhong, V. V. Bamm, M. A.M. Ahmed, 'Solid-state NMR spectroscopy of 18.5 kDa myelin basic protein reconstituted with lipid vesicles: Spectroscopic characterisation and spectral assignments of solvent-exposed protein fragments', Biochimica et Biophysica Acta, pp. 3193–3205, (2007).

[15] R. Dominguez, 'Actin-binding proteins – a unifying hypothesis', Trends Biochem. Sci, Vol.29, 572-5780, (2004).

[16] J. M. Boggs, I. R. Bates, A. A. Musse G. Harauz, 'Interactions Of The 18.5 Kda Myelin Basic protein With Lipid Bilayers: Studies By Electron Paramagnetic Resonance Spectroscopy And Implications For Generation Of Autoimmunity', In Multiple Sclerosis, Myelin Basic Proten, Editor Joan M. Boggs, (2008).

[17] http://www.python.org/, Python Software Foundation, (2012).

[18] C. M. Deber, C. Wang, L. L.P. Liu, A S. Prior, S. Agrawal, B. L. Muskat, A. J. Cuticchia, 'TM Finder: A prediction

program for transmembrane protein segments using a combination of hydrophobicity and nonpolar phase helicity scales', Protein Science, vol 10, pp.212-219, (2001).

Author Introduction

Massimo Mucciardi holds a PhD in Statistics from Messina University, Italy. He is an Assistant Professor in the School of Statistics at the University of Messina, Italy. His main research fields include spatial statistics, kernel methods, GIS and survey sample.

Cinzia Di Salvo holds a PhD in Biochemistry from Messina University, Italy. She has a grant at the University of Messina, Italy. Her main research fields regard protein purification and characterization.

Giovanni Pirrotta holds a PhD in Mathematics from Messina University, Italy. He is a software architect at the University of Messina, Department of Social and Human Sciences. His main research field includes Knowledge Representation, Semantic Web and Education/E-Learning Applications.

A Cloud-Based DSS Model for Cell Formation Problem

Chin-Chih Chang

Department of Information Management, Jen-Teh Junior College of Medicine, Nursing and Management

Miaoli, Taiwan

Email: chinju.chang@gmail.com

Abstract

In this study, we use the respective advantages of the tabu search (TS) and the cloud computing technologies to develop a cloud-based decision support system (DSS) for cell formation (CF) problem. To further verify the feasibility and effectiveness of the developed system, an example taken from the literature is adopted for illustrational purpose. Moreover, a set of test problems with various sizes drawn from the literature are used to test the performance of the proposed system. Corresponding results are compared to several well-known algorithms previously published. The results indicate that the proposed cloud computing CF DSS improves the best results found in the literature for 50% of the test problems. Moreover, with the assistance of our developed system, the CF practitioners in the production departments can interact with the systems without knowing the details of algorithms and can get the best machine cells and part families with maximize grouping efficacy wherever and whenever they may need it. These show that the proposed CF DSS should thus be useful to both practitioners and researchers.

Keywords

Cloud Computing; Cell Formation; Tabu Search; Decision Support System

Introduction

Cellular manufacturing (CM) is the application of group technology (GT) in manufacturing systems. GT is a manufacturing philosophy, which determines and divides the components into families and the machines into cells by taking advantage of part similarity in processing and design functions. Studies show that 30%–75% of the product cost is due to materials handling [1]. The implementation of cellular manufacturing has been reported to result in significant benefits such as reductions in material handling costs, work-in-progress inventory, throughput times and set-up times, simplified scheduling and improved quality [2].

Although CM may provide great benefits, the cellular manufacturing system (CMS) design is complex for real life problems. The foremost problem for CMS design is cell formation (CF), which groups the machines into machine cells and parts into part families. It has been known that the CF problem in CMS is one of the NP-hard combinational problems [3], as it becomes difficult to obtain optimal solutions in an acceptable amount of time, especially for large-sized problems. In this regard, many models and solution approaches have been developed to identify machine cells and part families. These approaches can be classified into three main categories [4]: (1) mathematical programming (MP) models, (2) heuristic/ meta-heuristic solution algorithms, and (3) similarity coefficient methods (SCM).

Due to their excellent performance in solving combinatorial optimization problems, meta-heuristic algorithms such as genetic algorithm (GA), simulated annealing (SA) and tabu search (TS) have been the most successful solution approach to efficiently solve the CF problem and its variants with good results. Among the meta-heuristic algorithms, TS has been successfully used to solve many problems appeared in manufacturing system including cell formation problems [5]. Hence, we adopt it as a solver to solve the CF problem in the development of our CF cloud-based decision support system (DSS).

With the rapid development of internet and the upgrades of software and hardware technologies, the concept of cloud computing is presented. The cloud computing can provide a number of services to users through the internet whenever and wherever. Moreover, due to an increasing global competition, companies are now shifting to a geographically distributed manufacturing environment. Besides, the information flow nowadays requires reliability, efficiency and security. With the emergence of information technology, the traditional way of communication of information flows between companies and between internal parties can now be replaced by the interconnected network [6].

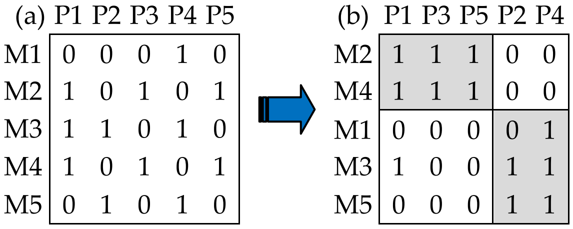
The Internet provides an open environment for companies to connect with their business partners as well as to serve as a medium for internal information flows [7]. Moreover, manufacturing systems have migrated to integrate with the internet to provide a remote access and control system with the characteristics of quick response and real line monitoring [8]. Thus, development of an effective computer-aided CF cloud-based DSS is necessary.

In this study, we use the respective advantages of the TS and the cloud-based technologies to develop a cloud-based DSS for CF problem. An example taken from the literature is adopted for illustrational purpose. To further verify the feasibility and effectiveness of the developed system, ten test problems with various sizes drawn from the literature are used to test the performance of our proposed CF solver. Corresponding results are compared to several well-known algorithms previously published.

The rest of the paper has been structured as follows: Section 2 describes the cell formation problem. Section 3 details the implementation of our cloud-based CF DSS. The performance of the proposed system and methodology is verified in Section 4 and the conclusion is given in Section 5.

Cell Formation Problem

Cell formation in a given 0-1 machine-part incidence matrix involves rearrangement of rows and columns of the matrix to create part families and machines cells, in which the cellular movement can be minimized and the utilization of the machines within a cell can be maximized. Two matrices shown in Figure 1 are used to illustrate the concept. **Figure 1(a)** is an initial matrix where no blocks can be observed directly. After rearrangement of rows and columns, two blocks can be obtained along the diagonal of the solution matrix in **Figure 1(b)**. For those 1's outside the diagonal blocks, they are called "exceptional elements"; while those 0's inside the diagonal blocks are called "voids".



(a)	P1	P2	P3	P4	P5
M1	0	0	0	1	0
M2	1	0	1	0	1
M3	1	1	0	1	0
M4	1	0	1	0	1
M5	0	1	0	1	0

(b)	P1	P3	P5	P2	P4
M2	1	1	1	0	0
M4	1	1	1	0	0
M1	0	0	0	0	1
M3	1	0	0	1	1
M5	0	0	0	1	1

FIGURE 1 REARRANGEMENT OF ROWS AND COLUMNS OF MATRIX TO CREATE CELLS: (A) INITIAL MATRIX AND (B) MATRIX AFTER REARRANGEMENT

There have been several measures of goodness of machine-part groups in cellular manufacturing in the literature. Two measures frequently used are the grouping efficiency [9] and the grouping efficacy [10] due to they are easy to implement. Grouping efficiency is defined as follows:

$$\eta = q\eta_1 + (1-q)\eta_2 \tag{1}$$

where η_1 is the ratio of the number of 1s in the diagonal blocks to the total number of elements in the diagonal blocks of the final matrix, η_2 is the number of 0s in the off-diagonal blocks to the total number of elements in the off-diagonal blocks of the final matrix, and q is a weight factor. Any 1s outside the diagonal blocks are called "exceptional elements," and 0s inside the diagonal blocks are called "voids."

Although grouping efficiency has been used widely, it was argued for its low discriminating capability in some cases affected by the size of the matrix. To overcome this problem, Kumar and Chandrasekharan [9] proposed another measure, the grouping efficacy, and can be defined as follows:

$$\Gamma = \frac{e - e_0}{e + e_v} \tag{2}$$

where e is the total number of 1's in the matrix, e_0 is the total number of exceptional elements, and e_v is the total number of voids. As grouping efficacy has been widely accepted in recent studies regarding CF problem, it is used as the performance measure for the proposed TS algorithm.

System Development

In this study, a cloud-based CF DSS is developed to get the best machine cells and part families with maximize grouping efficacy. The system architecture is shown in **Figure 2**. From the figure we can know that the system consists of five elements. They are the clients (i.e., users), the user interface, the web server, the CF solver and the database. All of them are linked up with the internet. We will describe them as follows:

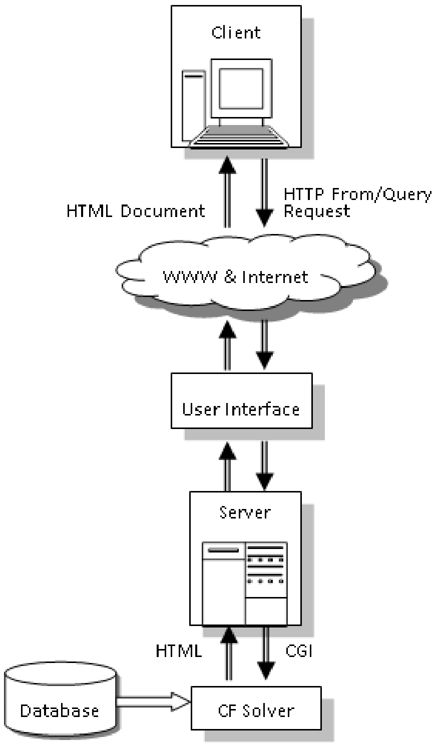


FIGURE 2 CLOUD-BASED ARCHITECTURE FOR CF DSS

Clients

web browsers are clients that connect to web servers and retrieve web pages for display. Using appropriate web browsers, such as Netscape's navigator or Microsoft's internet explorer, users can input data or view the CF results through a dynamic hypertext user interface.

User Interface

Because of the PHP is a widely-used general-purpose scripting language that is especially suited for web development and can be embedded into HTML. Hence, we use PHP to making dynamic and interactive web pages for the cloud computing user interface which consists of three buttons on the top of the screen. The framework of the user interface is shown in **Figure 3** which is simple and considered to be user-friendly. We will describe them below.

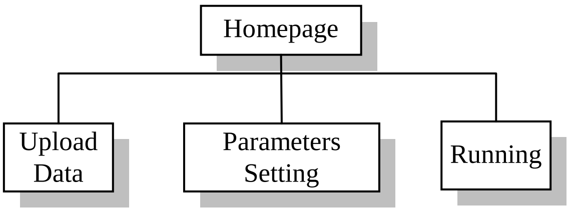


FIGURE 3 FRAMEWORK OF CLOUD COMPUTING USER INTERFACE

Web Server

The web server is a computer that serves requested web pages. The web server interacts with the individual user's web browser and accepts external Hypertext Transfer Protocol (HTTP) requests from the browser. An Application Programmer's Interface (API) is distributed, along with most of the commercially available browsers, such as Netscape's navigator or Microsoft's internet explorer. Application programs, such as PHP, ASP and JSP, can be written using these APIs to enhance the capabilities of a browser. Because of the Apache HTTP Server has been the most popular web server on the Internet since April 1996. Therefore, we use the apache server as web server in this study.

CF Solver

The CF solver was developed using Visual C++ programming languages. It consists of two stages: the initial solution construction and the improvements. The similarity coefficient-based method (SCM) is adapted in the first stage to produce good initial solutions, while the TS continuously improves and generates more effective solutions through the TS algorithm in the second stage. The proposed generic framework for the CF solver (HTSA) is shown in **Figure 4** which is actually consists of the following seven steps:

1) Initialization of computational parameters;

2) Construction of initial solution;

3) Searching of improving neighborhood solutions;

4) Update of tabu list;

5) Update of better solutions found;

6) Check of timing for directing searching toward diversified solution space by applying mutation operator;

7) Check of solution stagnancy.

Note that the first five steps are the same as the TS algorithm, while Step 6 generates new solutions with higher degree of diversification in order to increase the probability of finding the global optima, and Step 7 avoids spending too much computational efforts in order to have a balance between the computational effectiveness and efficiency.

For the CF solver, the insertion strategy is applied to produce a new neighborhood solution and the values of parameters are given below: tabu list size is equal to

7, a limit of iterations for each run is set to 1000 and the mutation probability for each gene is set at 0.8.

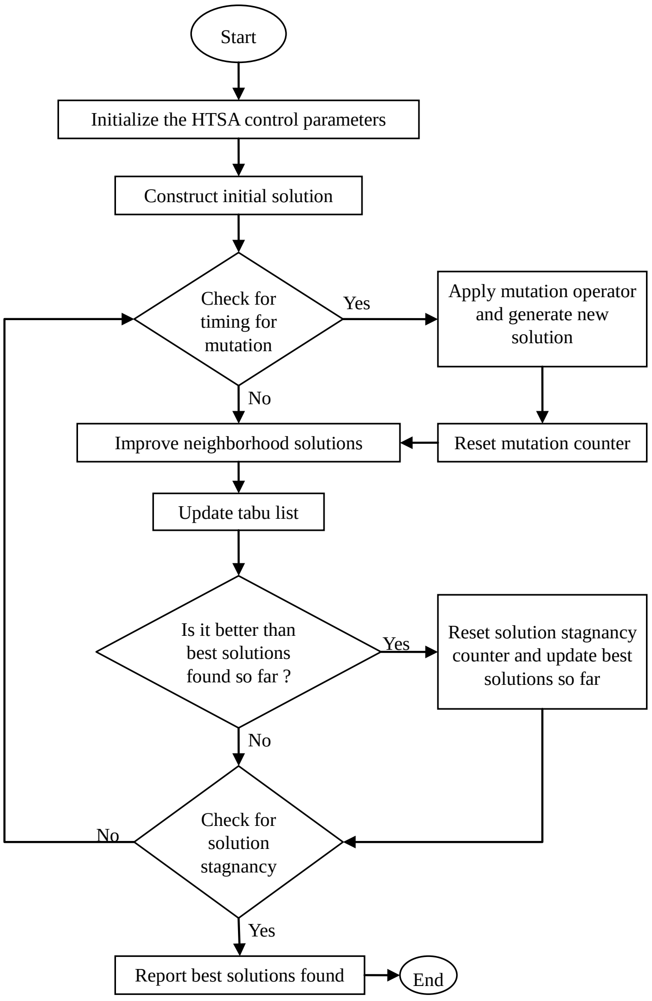

Start
Initialize the HTSA control parameters
Construct initial solution
Check for timing for mutation
Yes
Apply mutation operator and generate new solution
No
Improve neighborhood solutions
Reset mutation counter
Update tabu list
Is it better than best solutions found so far ?
Yes
Reset solution stagnancy counter and update best solutions so far
No
No
Check for solution stagnancy
Yes
Report best solutions found
End

FIGURE 4 FLOWCHART OF CF SOLVER (HTSA)

Database

A database server machine may be physically different from the web server that maintains the database. Due to the MySQL is the world's most popular open source database. Hence, we use MySQL server as the database server in this study. This remote database is accessed through the Open Database Connectivity (ODBC) gateway to insert, delete or update information in the database.

Research Results

The research results consist of two parts. They are the numerical example and the comparative study. We will describe them below.

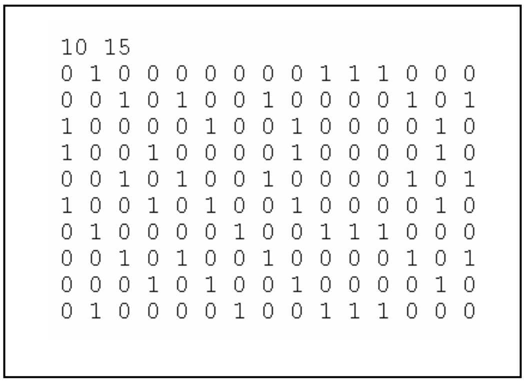
Numerical Illustration

We applied a numerical example, which was drawn from [11], to test the performance and usability of our developed system. The step-by-step procedures are described as follows:

Step 1: Press the "Upload Data" button to upload the 0-1 machine-part incidence matrix to web server, as shown in **Figure 5**, which consists of 10 machines and 15 parts.

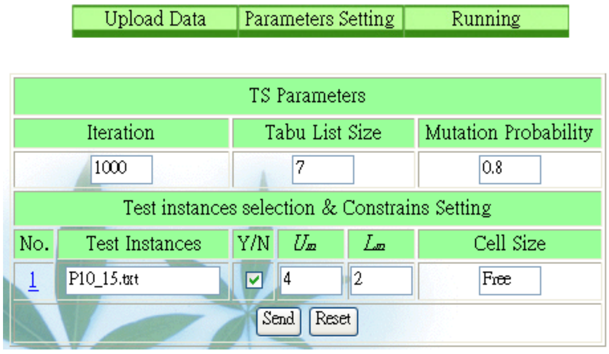
Step 2: Set the parameters and constrains for CF solver by pressing the "Parameters Setting" button, as shown in **Figure 6**, tabu list size is equal to 7, a limit of iterations for each run is set to 1000 and the mutation probability for each gene is set at 0.8. The maximum number of machines in each cell (U_m) is limited to 4 and the minimum number of machines in each cell (L_m) is 2.

Step 3: Press the "Running" button to execute the CF solver to groups the machines into machine cells and parts into part families with maximize grouping efficacy. As shown in **Figure 7**, the CF solver only took 0.0833 seconds to get the final 0-1 machine-part incidence matrix with three cells and 92% grouping efficacy.



```
10 15
0 1 0 0 0 0 0 0 0 1 1 1 0 0 0
0 0 1 0 1 0 0 1 0 0 0 0 1 0 1
1 0 0 0 0 1 0 0 1 0 0 0 0 1 0
1 0 0 1 0 0 0 0 1 0 0 0 0 1 0
0 0 1 0 1 0 0 1 0 0 0 0 1 0 1
1 0 0 1 0 1 0 0 1 0 0 0 0 1 0
0 1 0 0 0 0 1 0 0 1 1 1 0 0 0
0 0 1 0 1 0 0 1 0 0 0 0 1 0 1
0 0 0 1 0 1 0 0 1 0 0 0 0 1 0
0 1 0 0 0 0 1 0 0 1 1 1 0 0 0
```

FIGURE 5 0-1 MACHINE-PART INCIDENCE MATRIX



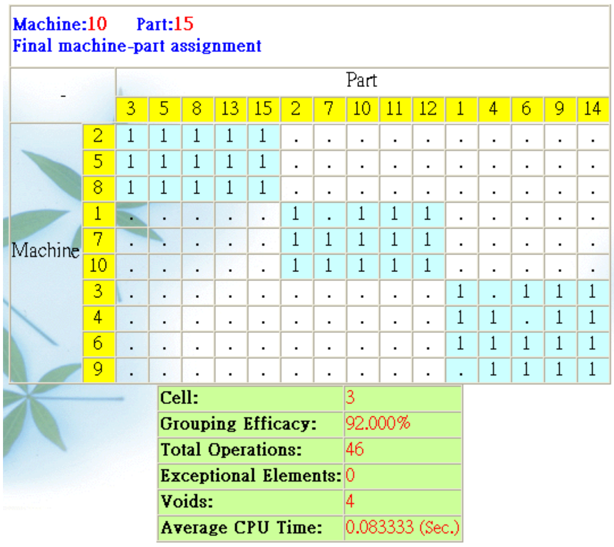
TS Parameters

Iteration	Tabu List Size	Mutation Probability
1000	7	0.8

Test instances selection & Constrains Setting

No.	Test Instances	Y/N	U_m	L_m	Cell Size
1	P10_15.txt	☑	4	2	Free

FIGURE 6 INPUT INTERFACE FOR SETTING PARAMETERS AND CONSTRAINS



Machine:10 Part:15
Final machine-part assignment

-		Part														
		3	5	8	13	15	2	7	10	11	12	1	4	6	9	14
Machine	2	1	1	1	1	1	.	.	.	.	.	.	.	.	.	.
	5	1	1	1	1	1	.	.	.	.	.	.	.	.	.	.
	8	1	1	1	1	1	.	.	.	.	.	.	.	.	.	.
	1	.	.	.	.	.	1	.	1	1	1	.	.	.	.	.
	7	.	.	.	.	.	1	1	1	1	1	.	.	.	.	.
	10	.	.	.	.	.	1	1	1	1	1	.	.	.	.	.
	3	.	.	.	.	.	.	.	.	.	.	1	.	1	1	1
	4	.	.	.	.	.	.	.	.	.	.	1	1	.	1	1
	6	.	.	.	.	.	.	.	.	.	.	1	1	1	1	1
	9	.	.	.	.	.	.	.	.	.	.	.	1	1	1	1

Cell:	3
Grouping Efficacy:	92.000%
Total Operations:	46
Exceptional Elements:	0
Voids:	4
Average CPU Time:	0.083333 (Sec.)

FIGURE 7 OUTPUT INTERFACE FOR DISPLAYING CF RESULTS

Comparative Study

In order to evaluate the computational characteristics of our proposed CF solver with other approaches, ten test instances from the literature, as shown in **Table 1** are used. The proposed CF solver are coded in Visual C++ programming languages and implemented on an Intel(R) 1.66 GHz personal computer with 1GB RAM. **Table 2** shows the comparisons of our proposed CF solver with other approaches from the literature, that is, the EA [12], the SA [13] and the HGA [14]. The bold characters indicate the best values obtained for each test problem. It can be seen from **Table 2** that our proposed CF solver are better than or equal to those reported results. To be more specific, CF solver obtains for 5 (50%) problems values of the grouping efficacy that are equal to the best results found in EA, SA, and HGA methods and improves the values of the grouping efficacy for the rest 5 (50%) problems. It is worth to mention that our proposed CF solver is able to find the optimal solution in 5.784 seconds, illustrating the superiority of CF solver in solution efficiency.

TABLE 1 SELECTED PROBLEM FROM THE LITERATURE

No.	Source	Size
1	King [15]	16×43
2	Carrie [16]	20×35
3	Chandrasekharan and Rajagopalan [17]	24×40
4	Chandrasekharan and Rajagopalan [17]	24×40
5	Chandrasekharan and Rajagopalan [17]	24×40
6	Chandrasekharan and Rajagopalan [17]	24×40
7	Chandrasekharan and Rajagopalan [17]	24×40
8	Chandrasekharan and Rajagopalan [17]	24×40
9	Stanfel [18]	30×50
10	Stanfel [17]	30×50

TABLE 2 PERFORMANCE OF THE PROPOSED ALGORITHM COMPARED TO OTHER ALGORITHM

Test instances		Other approaches			Proposed approach		
		EA	SA	HGA	CF solver		
No.	Size	Γ (%)	Γ (%)	Γ (%)	Cell Size	Γ (%)	CPU time (s)
1	16×43	54.86	52.44	54.86	8	**56.85**	0.975
2	20×35	76.22	78.40	76.14	5	78.40	0.409
3	24×40	100.00	100.00	100.00	7	100.00	0.844
4	24×40	85.11	85.11	85.11	7	85.11	0.987
5	24×40	73.51	73.51	73.51	7	73.51	1.013
6	24×40	51.88	52.44	52.50	11	**53.29**	2.200
7	24×40	46.69	47.13	46.84	12	**48.95**	2.794
8	24×40	44.75	44.64	44.85	12	**46.26**	2.828
9	30×50	59.21	60.12	59.66	13	60.12	4.272
10	30×50	50.48	50.51	50.51	14	**50.83**	5.784

Conclusions

In this study, we have used the respective advantages of the tabu search (TS) and the cloud computing technologies to develop a cloud-based decision support system (DSS) for cell formation (CF) problem. With the assistance of CF DSS, the CF practitioners in the production departments can interact with the system without knowing the details of algorithms and can get the best machine cells and part families with maximize grouping efficacy wherever and whenever they may need it. An example taken from the literature has been adopted for illustrational purpose and a set of test problems with various sizes drawn from the literature have been used to test the performance of the CF solver. The results have indicated that the proposed algorithm improved the best results found in the literature for 50% of the test problems and the CPU times for finding the optimal solution are in 5.784 seconds. These show that our developed system should be useful to both practitioners and researchers.

REFERENCES

[1] S. Heragu, Facilities Design, MA: PWS Publishing Company, Boston (1997).

[2] U. Wemmerlov and N. Hyer, Research issues in cellular manufacturing, International Journal of Production

Research. 25, (1987).

[3] A. Kusiak, Intelligent Manufacturing Systems, NJ: Prentice Hall, Englewood Cliffs (1990).

[4] Y. Yin and K. Yasuda, Similarity coefficient methods applied to the cell formation problem: A taxonomy and review, International Journal of Production Economics. 101, (2006).

[5] S. Lozano, B. Adenso-Diaz, and L. Onieva, A one-step Tabu search algorithm for manufacturing cell design, Journal of the Operational Research Society. 50, (1999).

[6] G.Y. Tian, G. Yin, and D. Taylor, Internet-based manufacturing: a review and a new infrastructure for distributed intelligent manufacturing, Journal of Intelligent Manufacturing. 13, (2002).

[7] J. Lee, E-manufacturing systems: fundamental and tools, Robotics and Computer- Integrated Manufacturing. 9, (2003).

[8] C.S. Smith and P.K. Wright, CyberCut: a World-Wide web-based design-to-fabrication tool, Journal of Intelligent Manufacturing. 15, (1996).

[9] M.P. Chandrashekharan and R. Rajagopalan, An ideal seed non-hierarchical clustering algorithm for cellular manufacturing, International Journal of Production Research. 24, (1986).

[10] C.S. Kumar and M.P. Chandrasekharan, Grouping efficacy: A quantitative criterion for goodness of block diagonal forms of binary matrices in group technology, International Journal of Production Research. 28, (1990).

[11] H. M. Chan and D. A. Milner, Direct clustering algorithm for group formation in cellular manufacture, Journal of Manufacturing System. 1, (1982).

[12] J. Goncalves and M. Resende, An evolutionary algorithm for manufacturing cell formation, Computers and Industrial Engineering. 47, (2004).

[13] T.H. Wu, C.C. Chang, and S.H. Chung, A simulated annealing algorithm to manufacturing cell formation problems, Expert Systems with Applications. 34, (2008).

[14] T. Adnan, H. Iftikhar, and G. Abdu, A hybrid genetic algorithm for machine-part grouping, Computers and Industrial Engineering. 56, (2009).

[15] J.R. King, Machine-component grouping in production flow analysis: An approach using a rank order clustering algorithm, International Journal of Production Research. 18, (1980).

[16] S. Carrie, Numerical taxonomy applied to group technology and plant layout, International Journal of Production Research. 11, (1973).

[17] M.P. Chandrasekharan and R. Rajagopalan, Groupability: an analysis of the properties of binary data matrices for group technology, International Journal of Production Research. 27, (1989).

[18] L. Stanfel, Machine clustering for economic production, Engineering Costs and Production Economics. 9, (1985).

A Completely Open-Source Finite Element Modeling Chain for Tubular Tissue-Engineered Constructs

Adrienne Madison and Mark A. Haidekker*

University of Georgia, College of Engineering, Athens, GA, USA

*Corresponding Author

Email: mhaidekk@uga.edu

Abstract

Finite-element modeling (FEM), well-established to predict the mechanical behavior of mechanical systems, enjoys growing popularity in the study of the biomechanical behavior of biological tissues. Contrary to mechanical systems, which can often be described analytically or with geometric primitives, biomedical objects require discretization of their often irregular shape. In noninvasive studies, imaging methods are used to obtain the shape. Frequently, a relationship between image intensity and biomechanical properties is assumed. Commercial FEM toolchains exist, but we failed to obtain a satisfactory discretization from simple phantom images. Driven by the application to image and characterize tissue-engineered blood vessels noninvasively, we sought to establish a completely open-source FEM toolchain. The open-source feature gives users the ability to modify and extend the code, and thus offers additional flexibility over commercial systems. We demonstrate that the combination of a custom module to discretize the geometry (meshing) combined with the open-source FEM solver *Tochnog* and free visualization software (namely, *Paraview* and *OpenDX*) completes an open-source FEM toolchain. We demonstrate its ability to analyze tubular phantoms modeled after tissue-engineered blood vessels and compare results to a commercial toolchain. We conclude that a fully open-source toolchain is feasible, but the critical element is the meshing module.

Keywords

Open-Source Software; Mesh Generation; Tissue Biomechanics; Vascular Grafts; Finite Element Modeling; FEM

Introduction

Finite-element modeling (FEM) has been intensely used in biomedical contexts to examine, simulate, and predict the material behavior and non-linear biomechanical properties of soft tissues through integration with medical imaging modalities [1]. Areas of application include the brain [2], lungs [3], and most importantly to us, vascular stress analysis [4-7]. The underlying principle of FEM is to subdivide the object of interest into a large number of connected small volumes that are considered homogeneous. These small volumes -- the finite elements -- are considered to be rigidly connected, thus providing boundary conditions imposed upon the element by its neighbors (forces, displacement) and by the environment (pressure, fixed and immobile elements, or moving elements).

One key challenge in the application of FEM in the biomedical context is the extraction of the object of interest from a medical image (i.e., segmentation) and the conversion of the object into a connected set of finite elements (meshing) that not only represents the geometry of the object, but also the approximate local material properties in the tissue section that corresponds to the finite element. Only when this information is available -- the geometry, discretized into finite elements, and the associated material properties -- the response of the tissue to external mechanical loads and the pressure can be examined. Algorithms capable of constructing triangular and tetrahedral-shaped FEM meshes from medical images exist [8,9], yet neither study addresses the issue of segmentation. In fact, segmentation is usually considered a separate step, although one pre-processing toolkit [10] specializes in combining segmentation and meshing of 3D models. The algorithms implemented are also capable of material property assignments. Another related software toolkit reported in [11] provides the assignment of both material properties as well as boundary/loading conditions.

The driving application behind the toolchain presented in this manuscript is the recent development of fully biological tissue-engineered blood vessels [12,13]. Blood vessels that are grown from the patient's own cells must be tested exhaustively on an individual sample basis because of the inter-sample variation. Our vision is to use an imaging method, such as optical coherence tomography (OCT), confocal imaging [14,15], or optical trans-illumination tomography [16], then use the resulting image data to generate a FEM mesh that allows the prediction of its biomechanical behavior when exposed to pulsatile blood pressure. Studies exist where FEM was applied on vascular entities, such as arteries and veins [4-7, 17-19]. Commonly, model geometries, based on measurements obtained from real vessels, were constructed with commercial CAD and FEM software. However, these studies only focus on the results obtained from the analysis, and not specifically on the processes to prepare 3D image data for analysis; detail information in those publications was accordingly sparse.

Several finite element solvers are available from different companies. We initially used a departmental license of Algor

(Algor, Inc., Pittsburgh, PA), a combination of FEM simulation software with limited visualization features. The generation and analysis of analytically-defined cylindrical models was straightforward, and no fundamental challenges were encountered. However, it was not possible to manually generate the complex and irregular shapes of a sample blood vessel. A separate software program was necessary to extract the mesh from the original image. Few such programs are offered, because engineering design often allows to describe mechanical objects in an analytical fashion or as a combination of geometrical primitives. One software program that allows to generate a mesh from medical images is Mimics (Materialise, Leuven, Belgium). We prepared a simple phantom **(Figure 1)** to test the ability of Mimics to generate a mesh that can be processed by Algor. In spite of extensive support from Materialise, we were not able to generate such a mesh. The Mimics meshes showed irregular element boundaries and shapes, and Mimics produced a very large number of nodes that could not be reduced and that caused Algor to fail. We concluded that the combination of Mimics and Algor would not meet our requirements. To avoid paying the license fees for additional software, we decided to attempt the design of a FEM chain based entirely of free open-source software for use in the analysis of tubular constructs.

Materials and Methods

Finite-element modeling of tissue engineered blood vessels is generally performed using some form of volumetric image obtained, for example, by computed tomography (CT), magnetic resonance imaging (MRI), or -- on a smaller scale -- by optical coherence tomography (OCT) and involves four main steps. In the first step, the object of interest is separated from the image background (segmentation). The meshing step follows, during which the segmented object is subdivided into small elements, and boundary conditions and material properties are applied to the elements. The third step involves the actual finite element simulation of the time-variable behavior of the object under the established parameters. Once this is completed, the output of the simulation process is visualized or post-processed in the final step.

Phantom Generation

Two phantoms were used throughout this study that approximated pre-segmented blood vessels when scanned in air with computed tomography. Both phantoms were embedded in a 256 by 256 by 128 voxel matrix. Voxels were anisotropic with a size of approximately 23 microns in the x-y-plane and 92 microns in the axial (z) direction. The first phantom, shown in **Figure 1**, was modeled after an autologous vascular graft described by L'Heureux et al. [13]. The vascular graft has typical dimensions of 5.0 x 10^{-4} m (0.5 mm) thickness, 4.0 x 10^{-2} m (40 mm) length, and 4.0 x 10^{-3} m (4 mm) diameter, which is closely matched by our phantom, except that the length was truncated to approximately 1.2 x 10^{-2} m (12mm).

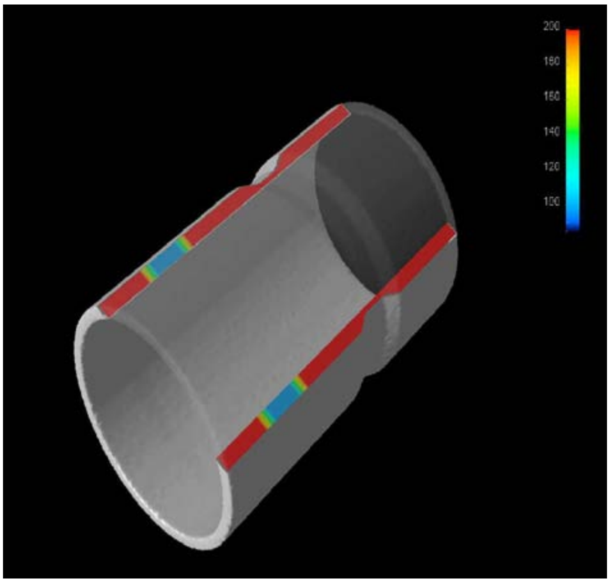


FIG. 1 3D RENDERING OF THE FIRST PHANTOM USED IN THIS STUDY.THE PHANTOM IS A TUBULAR OBJECT THAT IS WIDELY HOMOGENEOUS, BUT HAS TWO INHOMOGENEOUS REGIONS. THE FIRST REGION HAS A THINNER WALL, WHEREAS THE SECOND REGION HAS LOWER IMAGE VALUES. IN THIS RENDERING, THE OBJECT HAS BEEN CLIPPED, AND THE CUT SURFACES HAVE BEEN FALSE-COLORED TO HIGHLIGHT THE CHANGE IN DIAMETER AND CHANGE IN IMAGE VALUES.

The first phantom was widely homogeneous, but featured two annular inhomogeneities, one section where the wall thickness was reduced, and one section where the image intensity was lowered, while the wall thickness was held constant. The first section was created to simulate the fixation grooves (visible in **Figure 2**), and the second section represented incompletely fused tissue layers. This second section translates into different material groups. For comparison, the cross-section of a CT image of an actual tissue-engineered blood vessel is shown in **Figure 2**.

The second phantom was modeled after a fusiform aneurysm in a curved section of blood vessel. The outer dimensions, that is, 4.0×10^{-3} m diameter, 5.0×10^{-4} m wall thickness and 1.2×10^{-2} m length were kept similar to the first phantom, but a 15° bend was introduced, and the central region featured an elliptical expansion, with the inner circular region filled with a simulated plaque, i.e., a region with higher material stiffness. A rendering of the second phantom is shown in **Figure 3**.

Mesh Extraction Module

We decided to develop a relatively simple mesh extraction module based on the principle of radial probing rays [20] that seamlessly interfaces with the chosen open source FEM software, uses the near-cylindrical geometry, and may serve as a demonstration model for more advanced meshing modules.

The input of the module is assumed to be pre-segmented, that is, all background voxels have a zero value, and all voxels that belong to the vessel phantom have a nonzerovalue. In a first pass over the image, the centroid for each slice is determined and a straight line fitted into these centroids. This line is a first-order approximation of the lumen center capable of reflecting a tubular object that is tilted or slightly curved.

In the second pass over the sample, radial lines are emitted in each slice from the point where the central line intersects the slice (**Figure 4**). Starting from this point, image values are sampled along each ray. Once a nonzero image value is encountered for the first time, this point is marked as a node on the inner wall. Tracing continues until the image values drop to zero again, and this point is marked as a node on the outer wall. Thus each ray provides one pair of nodes. Two adjoining rays together with their corresponding rays in the next slice provide eight nodes, and these eight nodes define one finite element. The relationship of the rays and nodes to one element is shown in **Figure 5**. At the same time, image values inside the approximated cuboid are averaged to provide a material index for the respective element. After user-selectable binning, the element can be assigned to a material group, and Young's modulus for each material group is user-specified.

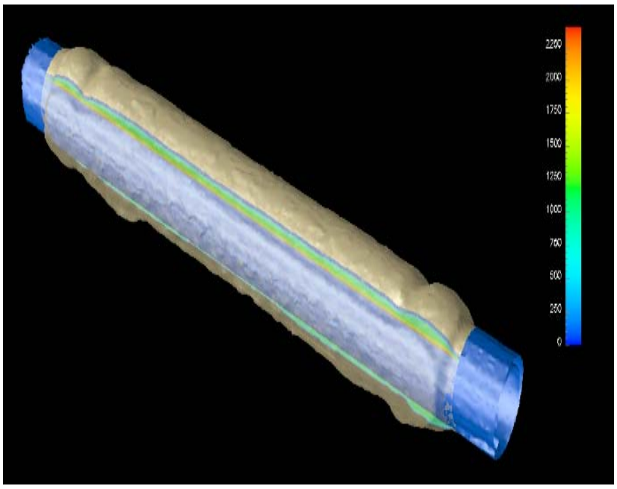


FIG. 2 3D RENDERING OF A TISSUE-ENGINEERED BLOOD VESSEL [13] OBTAINED BY COMPUTED TOMOGRAPHY.THE TISSUE LAYER (INDICATED IN AN OFF-YELLOW COLOR) IS GROWN ON A STEEL MANDREL (BLUE) WITH 4 MM OUTER DIAMETER. THE TISSUE COMPLETELY ENCLOSES THE MANDREL, BUT THE TISSUE WAS CLIPPED IN THIS IMAGE TO SHOW THE CROSS-SECTIONAL INTENSITY DISTRIBUTION AND THE THICKNESS IRREGULARITIES. THE FIXATION GROOVES SHOW PROMINENTLY AT EACH END OF THE TISSUE SECTION. NOTE THAT THE APPARENT TISSUE DENSITY INCREASE NEAR THE MANDREL IS AN ARTIFACT CAUSED BY PARTIAL-VOLUME EFFECTS.

In a third pass over the now-generated surface discretization, mesh refinement by interpolation is possible. When interpolation is selected, a user-defined number of nodes are interpolated between two adjoining rays by means of natural cubic splines.

The meshing module alternatively outputs a STL (stereo lithography) file for immediate visualization, or a Tochnog input/control file. The Tochnog file structure, together with an explanation of the required sections for the FEM software, is provided in the Appendix.

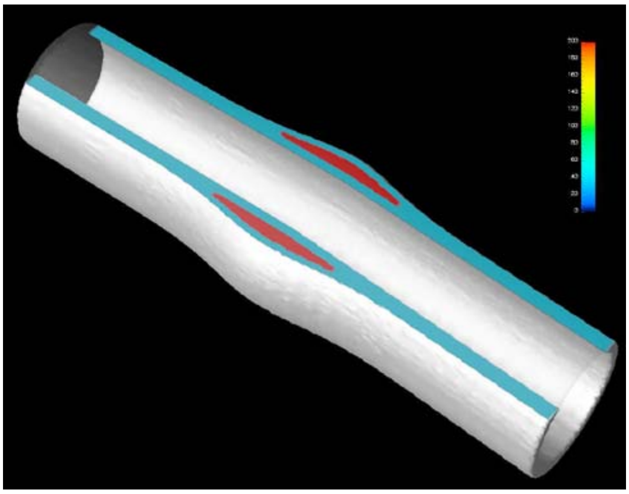


FIG. 3 3D RENDERING OF THE ANEURYSM MODEL FOR THE SECOND PHANTOM.THE MODEL REPRESENTS A FUSIFORM ANEURYSM ALONG A BENT BLOOD VESSEL WITH A 15°CURVE. THE FALSE-COLORED IMAGE VALUES REPRESENT THE MATERIAL ELASTICITY, IN THIS CASE, LIGHT BLUE FOR THE REGULAR VESSEL WALL AND RED FOR THE MORE RIGID PLAQUE. THE COLOR SCALE BAR IS THE SAME AS IN **FIGURE 1.**

FEM Software

A mature FEM solver that is available under a free open-source license exists. Tochnog, an implicit/explicit solver for a number of different problems, including linear and nonlinear solids, fluids and gases (Navier-Stokes), and for solving the wave equation, can be found at (tochnog. sourceforge. net). The output file generated by the mesh extraction module directly serves as a Tochnog input/control file, and only minimal editing (specifically, modification of the material definition) is needed. Control statements within the Tochnog input file determine its output format. At the end of the simulation run, the output file contains the node locations, node pressure, temperature, stress, strain, and velocity for each time step. Tochnog generates output files that are directly compatible with various post-processing and visualization packages.

Visualization

The visualization program GiD (CIMNE, Barcelona, Spain) is recommended by Tochnog. GiD is available without a license fee, but the no-cost version comes with a time limit. We examined both Paraview (Kitware, Inc., Clifton Park, NY) and OpenDX (www.opendx.org) as alternative visualization modules at the end of the FEM processing chain. Paraview accepts many input formats, and we used the VTK (visualization toolkit) format to exchange data between Tochnog and Paraview. OpenDX requires a specific input format, which Tochnog can provide. In addition, the meshing module can produce an STL file, which is a tesselated description of the object's surface without any material properties. Many STL file viewers are available, and we used gmsh (http://geuz.org/gmsh/).

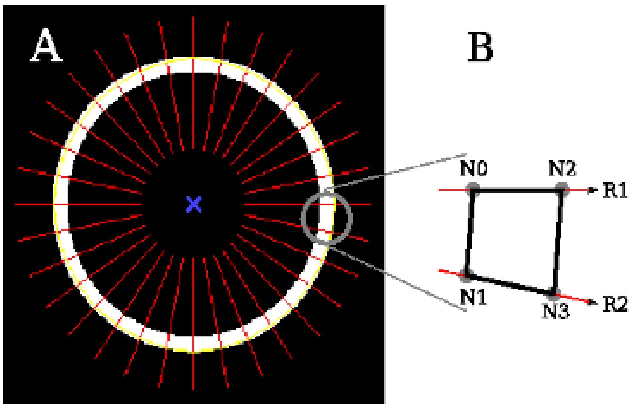


FIG. 4 EXTRACTION OF THE BOUNDARIES OF A CONVEX, TUBULAR OBJECT WITH PROBING RAYS (A).IN EACH SLICE, PROBING RAYS ARE EMITTED FROM THE CENTROID (BLUE X-MARK) AT REGULAR, ADJUSTABLE ANGULAR INTERVALS. IMAGE VALUES ARE SAMPLED ALONG THE PROBING RAYS. WHEN A PRE-SELECTED THRESHOLD IS FIRST EXCEEDED, AN INTERSECTION (NODE) OF THE RAY WITH THE INNER WALL IS RECORDED. ONCE THE IMAGE VALUES DROP BELOW THE THRESHOLD AGAIN ALONG THE RAY, THE INTERSECTION OF THE RAY WITH THE OUTER WALL IS RECORDED. TWO SUBSEQUENT RAYS (R1 AND R2) THEREFORE DEFINE A QUADRILATERAL, WHICH IS ONE FACE OF AN ELEMENT. A MAGNIFIED SECTION (B) SHOWS THE NODES. THE NUMBERING OF THE NODES CORRESPONDS TO FIGURE 5, AND NODES NEED TO BE ORDERED AS INDICATED BY THE NODE NUMBERS.

Results

Proof of Principle

To test the overall FEM chain from the initial volumetric image to the final visualization, we assumed that our phantom model (**Figure 1**) was a thin-walled, relatively elastic blood vessel being subjected to an internal pressure that corresponds to blood pressure acting on the inner lumen of the vessel. We also assumed that the ends of the vessel would be fixated (fixation grooves in **Figure 2**). Initially, we used 60 probing rays per slice and one set of rays every slice for a total of 15,360 nodes and 7,620 elements. The ray values were selected such that the division of the number of rays by the 360 degrees in a circle would result in a whole number multiple of 360.

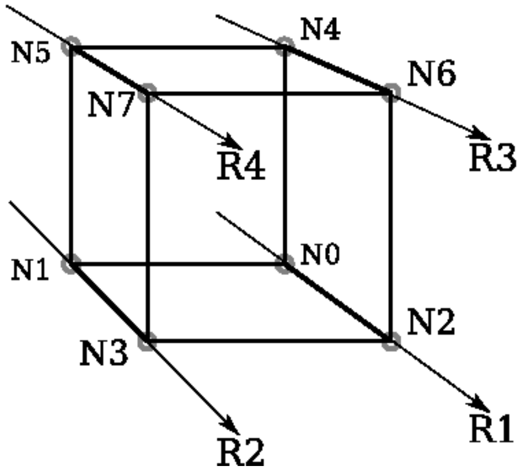


FIG. 5 RELATIONSHIP OF THE PROBING RAYS (R1 THROUGH R4) TO THE NODES AND FACES OF AN ELEMENT. THE LOWER TWO RAYS (R1 AND R2) BELONG TO SLICE Z, WHEREAS THE UPPER TWO RAYS (R3 AND R4) BELONG TO THE SUBSEQUENT SLICE AT Z+ΔZ. FROM THE OBSERVER'S POINT OF VIEW, RAYS ARE PROCESSED FROM RIGHT TO LEFT. THE NODES (INDICATED BY GRAY CIRCLES) ARE ARRANGED IN A ZIGZAG PATTERN WHERE THE FIRST TWO NODES (N0 AND N1) LIE ON THE INSIDE WALL, AND THE NEXT TWO NODES (N2 AND N3) LIE ON THE OUTSIDE WALL, WHEREBY THE CONNECTING VECTORS N0→N1 AND N2→N3 BOTH POINT RIGHT-TO-LEFT. THE SAME ORIENTATION IS USED FOR THE NODES IN THE UPPER SLICE, N4 THROUGH N7.

Using an in-plane pixel size of 2.3 x 10^{-5}m, we subjected the internal nodes to a pressure of -16 kPa (approximately 120 mmHg). The negative sign indicates outward-directed pressure. Each end of the tube was fixed by labeling them as boundary nodes and restricting movement in the x-, y-, and z-directions. We selected a Poisson ratio of 0.45, and a density of 945 kg/m^3. Based on the average intensity values calculated, five material groups were generated possessing elastic moduli values of 55, 60, 75, 85, and 100 MPa which were assigned and chosen to make the construct relatively elastic. All of the mechanical property values were selected such that our model could be characterized as having a rubber-like material composition.

The visual results of subjecting our phantom model to our modeling chain, created with GiD, are presented in **Figure 6**. The images depict the generated mesh onto the 3D tubular model, the classification of material groups present, as well as an exaggeration of the deformation behavior resulting from pressure being applied to the inner wall.

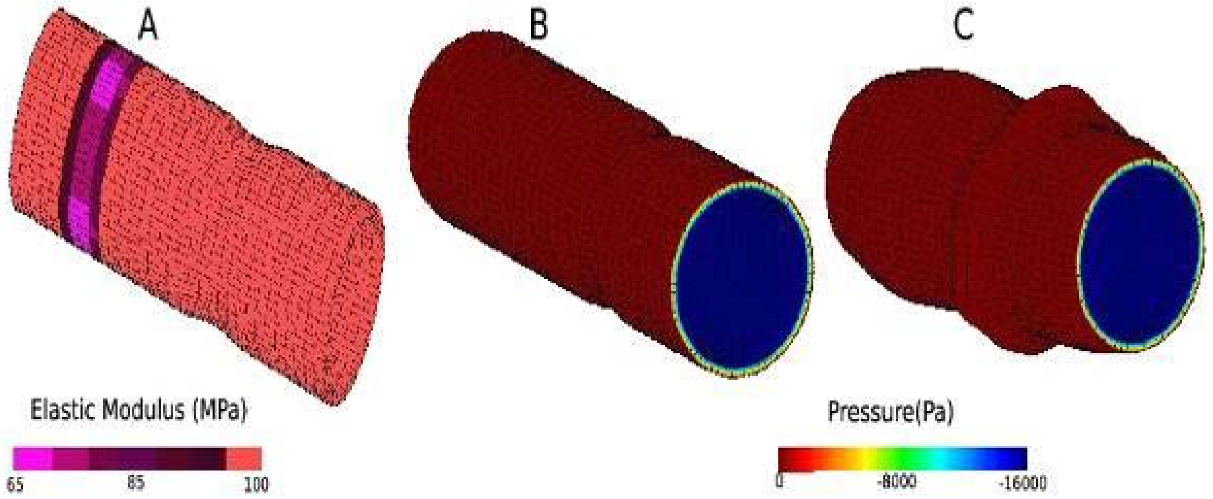


FIG. 6 GID VISUALIZATION RESULTS OF PHANTOM MODEL SUBJECTED TO HOMOGENEOUS INTERNAL PRESSURE. THE PHANTOM IS DECOMPOSED INTO 15,360 NODES AND 7,620 ELEMENTS. THE CIRCULAR SECTION WHERE THE WALL IS THINNER AS WELL AS THE CIRCULAR SECTION WHERE IMAGE INTENSITY IS LOWER BECOME VISIBLE. REDUCED IMAGE INTENSITY IS REFLECTED IN MULTIPLE MATERIAL GROUPS (A). THE INTERNAL PRESSURE EXPANSION AND DISTRIBUTION BEHAVIORS OF THE THIN WALL AREA, LOWER INTENSITY AREA, AND ENDS OF TUBE ARE HIGHLIGHTED (B). THE EXAGGERATED SHAPE DEFORMATION RESULTING FROM THE INTERNAL PRESSURE EXPANSION IS ALSO SHOWN (C).

Mesh Convergence

We examined possible convergence behavior by varying (1) the number of nodal interpolation points between rays and (2) the ray distance in the *z*- (axial) direction. The material stress values from each analysis were chosen as the basis of the convergence observations. The results are presented in **Table 1**. In the case of (1), we used spline interpolation to generate additional nodes without changing the nodes of the non-interpolated reference mesh. For these experiments, 18 probing rays per slice were implemented because the angular increments of 20° provide adequate spacing for the addition of interpolated points between each pair of rays. Analyses were conducted on the tube that resulted in the addition of 0-8 nodes in between pairs of rays at each slice. The maximum stress values obtained at each interpolation level are similar, with maximum deviations of -2% and +3% from the mean value of 19.4 kPa. The axial anisotropy is the product of voxel anisotropy and the number of slices skipped for the mesh generation. For the geometry defined for the phantom, voxels are 4 times longer in the axial direction than in the x-y-plane. In the case of varying ray distance in the axial direction, two analyses were conducted at 18 rays without the addition of interpolated nodal points: (a) 18 rays per slice with one set of rays every four slices and (b) 18 rays every four slices, resulting in axial anisotropies of 4 and 16 respectively. We observed that number of nodes and the number of elements increase proportionally with the number of interpolated points; however, they decrease proportionally as the axial anisotropy increases. While the number of material groups are not affected by the number of rays and nodal and elemental amounts, increasing the axial anisotropy also decreases the number of material groups present. The internal wall pressure and expansion behavior remains unaffected by changes in nodal and elemental amounts. Lastly, the maximum stress values obtained at the higher axial anisotropy of 16 and in our proof-of-principle experiment using 60 rays (19.5 kPa) align with the values with presented in the table obtained using both a lower anisotropy of 4 and lower number of rays.

An out-of-the-box installation of Tochnog does not have multiprocessor capabilities, but a typical file with approximately 16,000 nodes takes only a few minutes to compute on a single CPU core. The amount of computational time required increases approximately linearly with increasing nodal and elemental amounts and decreases accordingly when the axial anisotropy increases. However, Tochnog can be linked with libraries that are multi-thread-capable, and execution time is reduced correspondingly.

Evaluation of the Simulated Aneurysm Geometry

In addition to the cylindrical geometry of the first phantom, we examined the performance of our meshing module in cases of more complex tubular geometries. Aneurysms are balloon-like bulges that occur in blood vessels as the inner blood vessel wall weakens and eventually tears as a result of high blood pressure. Blood, and in some instances plaque, begins to pool in localized positions between these weakened vessel layers. Continued growth or expansion of the bulge increases the potential for the vessel to rupture, leading to a host of other complications including hemorrhaging, stroke, and in extreme cases, death.

We decided to simulate a fusiform aneurysm in which a bulging deformation is visible on both sides of the vessel **Figures 3 and 7A**. In addition to its expanded section in the center with two different material properties, there is a height-dependent offset of the vessel geometry in the x-direction, resulting in a 15° angle between the medial axes of the two ends. The purpose of this kink is to prove that the meshing algorithm can accurately capture objects that are not fully cylindrical. With 60 probing rays per slice, the model is composed of a total of 15,360 nodes and 7,620 near-cuboid elements. The in-plane pixel size, pressure, Poisson ratio and density values were the same as outlined in Section 3.1. Based on the average intensity values calculated, eleven material groups were generated possessing elastic moduli values of 100, 105, 110, 115, 120, 125, 130, 135, 140, 145, and 150 Mpa. Elastic moduli with larger values are expected to be assigned within the bulging areas of the vessel. Different material groups occur when the element encloses different amounts of the two materials, an effect similar to partial-volume artifacts in computed tomography.

Figure 7B displays the material stress obtained from the Tochnog analysis. As expected, the highest material stress values are observed at the weakest areas of the inner vessel walls, where the bulging between layers is first observed. The areas of the vessel perpendicular to the bulges are believed to also

TABLE 1 EFFECTS OF AXIAL AND RADIAL MESH REFINEMENT ON THE TOTAL NUMBER OF NODES AND ELEMENTS, MAXIMUM STRESS VALUES, AND NUMBER OF MATERIAL GROUPS, AND FEM SOLVER EXECUTION TIME FOR THE TUBULAR PHANTOM.

Number of Rays	Axial Anisotropy	Interpolation	Nodes	Elements	Material Groups	Maximum Stress (kPa)	Time (min:sec)
18	4	0	4608	2286	5	19.0	0:22
18	4	1	9216	4572	5	19.1	0:45
18	4	2	13824	6858	5	19.2	1:10
18	4	3	18432	9144	5	19.8	1:32
18	4	4	23040	11430	5	19.0	1:55
18	4	5	27648	13716	5	19.2	2:24
18	4	6	32256	16002	5	20.0	2:48
18	4	7	36864	18288	5	19.6	3:17
18	4	8	41472	20574	5	19.9	3:45
18	16	0	1152	558	2	19.6	0:13

be subjected to increased stress due to the shape deformation (bulging) under high pressure and difference in material property values in comparison to the bulge.

Visualization Software Comparison

The images presented in **Figure 6** were constructed using GiD; however to adhere to our goal of presenting a completely open-source modeling chain, we compared the visualizations from GiD with those from Paraview and OpenDX. Paraview was found to provide visualizations comparable to GiD, whereby out-of-the-box visualization schemes provided immediate results with a low level of user input. A Paraview rendering example is presented in **Figure 7**. Conversely, OpenDX allowed to design complex visualizations with high flexibility. One sophisticated example is shown in **Figure 8**. However, OpenDX requires the generation of a visual program, and experience with OpenDX is a prerequisite. A unique feature of GiD is the ability to exaggerate deformation (prominently visible in **Figure 6C**). A comparable feature was not found in Paraview.

Lastly, we wanted to compare the performance of our meshing module to the mesh previously obtained by the closed-source medical image pre-processing software. We used Mimics to obtain the 3D mesh of our phantom model shown in **Figure 1**. This mesh was then imported into the commercial FEM software Algor for visualization purposes. The resulting mesh consisted of a mix of approximately 300,000 hexagonal, wedge, pyramid, and tetrahedral shaped elements and comprised approximately 100,000 nodes and 13,000 surfaces. **Figure 9** displays the visual results obtained from the Mimics meshing process. The simulation performed with Algor did not converge and did not produce any output.

Discussion

In this study, we presented a fully open-source toolchain for finite-element modeling. A number of reasons make such a toolchain an attractive alternative to commercial software packages. First and foremost, no license fees are incurred, and FEM analysis becomes possible on a low budget. Furthermore, the usability of the software for a specific purpose can be examined without obligation. The second fundamental advantage lies in the open-source nature of the software, which means that the underlying program code can be examined or modified. In a classroom setting, where FEM is often taught with black-box software under an academic license, students can use open-source software to examine the numerical aspects of solving discrete partial differential equations. In a research setting, the exact algorithm that leads to a specific result can be determined, potential weaknesses identified, and the code amended. Access to the algorithm is particularly important when critical nonlinear cases are examined, such as turbulence or fracture. On the other hand, open-source software generally has no organized support. Rather, this type of software relies on community support, and software-related questions are usually resolved by peers in Internet forums. Software development is also driven by community efforts, although sometimes (as in the example of Paraview) a company supports development. The centerpiece of our toolchain, the FEM solver Tochnog, is offered both as a free version and as a commercial version with paid support and development (Tochnog Professional, Feat, The Netherlands).

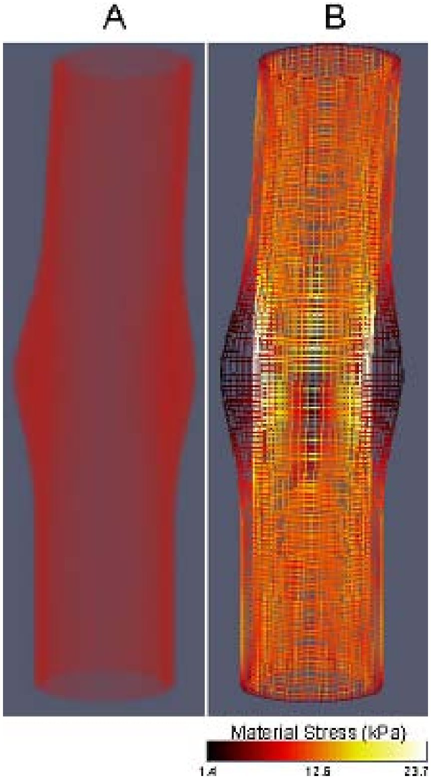


FIG. 7 PARAVIEW VISUALIZATION RESULTS OF THE FUSIFORM ANEURYSM PHANTOM MODEL SUBJECTED TO HOMOGENEOUS INTERNAL PRESSURE. COMPOSED OF 15,360 NODES AND 7,620 ELEMENTS, THE VOLUMETRIC GEOMETRY IS SHOWN (A) ALONG WITH THE INTERNAL AND EXTERNAL STRESS DISTRIBUTION THROUGHOUT THE VESSEL (B).

If sufficient demand exists, community-driven software packages of similar functionality can frequently be found. The visualization toolkit VTK, for example, has led to the development of Paraview, VisIt (Lawrence Livermore National Laboratory, Livermore, CA), and Slicer (www.slicer.org), to name a few examples. Alternatives to Tochnog exist as well, for example, FreeFEM (www.freefem.org) and Elmer (elmer.sourceforge.net). However, we found no free alternative to Mimics to create a mesh from a 3D medical image. This gap prompted us to develop a specialized meshing program for our research application of vascular constructs, and the implications of this gap are discussed in detail below.

Meshing Module

The most crucial step in all FEM chains that process medical images is the actual surface or volume parametrization. In simple terms, one could describe this step as converting a stack of pixels into an ordered set of nodes and elements. This step consists of two successive parts: Image segmentation and the actual surface parametrization. Interestingly, few commercial software packages and no open-source packages exist that perform this task. Mimics by Materialise is arguably the most widely used mesh generation package for medical images. Our experiments with Mimics led to meshes with a large number of irregular shapes, jagged edges and sharp drop off points. We observed that downstream software had difficulties processing such a mesh. Since we were unable to find the cause of those difficulties in spite of extensive help from Materialise support, we decided to favor the simplest possible geometry for our application and develop our own mesh extraction software with the key difference that, unlike Mimics, our software will be restricted to a limited set of geometries.

A second notable difference to Mimics is the ability of our module to automatically assign material parameters that correspond to varying image intensity values. Such a function does not exist in Mimics, which assumes a single homogeneous material. In addition, the adjustment of probing ray density, spline-based surface smoothing, and axial anisotropy adjustment features of our meshing module do not have a known correspondence in Mimics. These examples illustrate a typical trade-off between software aimed at solving as broad a range of problems as possible (Mimics) and software aimed at solving specialized cases (our meshing module).

In comparison with other FEM software chains, the uniqueness of our approach lies in the completely open nature of all steps of mesh generation, solving of the equations, and visualization. Other published partly open-source chains require the generated model to be imported and analyzed with commercial closed-source software [10, 11], or, in other cases, the meshing algorithms examined require the use of commercial software to both pre-process the image slices into a 3D model and undergo FEM analysis and do not appear to be open-source [8, 9]. In addition, most of these have only been implemented on anatomical bone structures, which are by nature more easily segmented.

Our meshing software extracts geometry from the image and applies a highly regular grid of nodes to the inner and outer wall. In addition, the software is capable of applying material properties and boundary/loading conditions to the model. Using the probing ray principle, the module is able to automatically generate finite cuboids, trapezoids, or frustums in a Tochnog-ready input file. Through calculation of average intensities, we were able to distinguish between inhomogeneities along the model and assign unique material definitions on an element-by-element basis under the assumption that a relationship between image intensity and material properties exists [21-24]. This assumption has also been implemented in other FEM analyses of medical images [3, 10]. This relationship is often empirical and highly dependent on the imaging modality used. In some examples, notably bone imaged by computed tomography, a strict relationship between the CT value and bone mineral density exists. Optical modalities can relate scattering to the presence of collagen [15], which is fundamental to tissue elasticity. In other cases (e.g., T1 and T2 relaxation in MRI), the relationship is more tenuous and needs to be established beforehand in separate studies. For the proof of principle that is the focus of this study, we used somewhat arbitrary values that are common for many elastomers, and we are aware that the precise model of a tissue-engineered blood vessel requires additional research, for example, a study with optical tomography [16], where the blood vessel is subjected to pressure and its expansion measured.

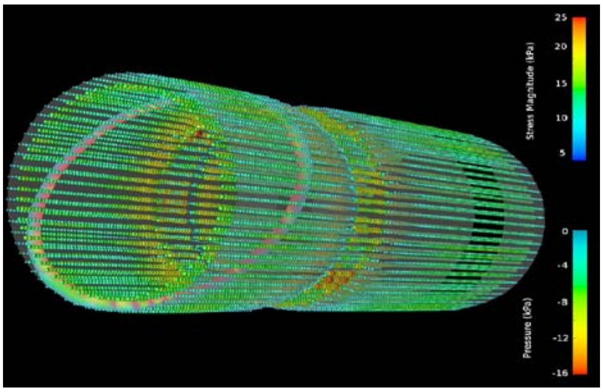


FIG. 8 EXAMPLE 3D RENDERING OF THE FEM SIMULATION RESULTS WITH OPENDX. FOR THIS EXAMPLE, THE INNER AND OUTER SURFACE WERE RENDERED AS GRAY, SEMI-TRANSPARENT TUBES. THE MAGNITUDE OF THE SHEAR STRESS TENSOR WAS SUPERIMPOSED FOR EACH NODE AS A GLYPH, I.E, A SMALL SPHERE WHERE THE SIZE IS PROPORTIONAL TO THE STRESS MAGNITUDE. THE GLYPHS ARE FALSE-COLORED WITH THE MAGNITUDE FOR IMPROVED VISUAL PERCEPTION AND THE VALUES REPRESENTING THESE COLORS ARE DISPLAYED IN THE UPPER COLORBAR. IN ADDITION, A SLANTED RING WAS PLACED INSIDE THE VESSEL WALL (NEAR THE LEFT END OF THE TUBE) THAT DISPLAYS THE MAGNITUDE OF THE PRESSURE BY USING THE LOWER COLORBAR (NEGATIVE VALUES INDICATE OUTWARD-DIRECTED PRESSURE).

In its present form, our meshing module is limited to approximately cylindrical, tubular objects. Concavities, for example, saccular aneurysms, would not be captured correctly by our module. Furthermore, the relatively coarse subdivision into one single element across the wall may cause some artificial rigidity. However, Tochnog performs automatic mesh refinement based on the residuals of the governing equations. Furthermore, our algorithm can be refined in a very straightforward manner with multiple thresholds. Presently, a binary image is assumed with one object of approximate radial homogeneity. If this assumption leads to an unacceptable simplification, a node along the ray can be created whenever one of multiple thresholds is crossed. This process may generate new shapes, most notably, triangular prisms and hexahedrons, for which different element definitions exist in Tochnog.

A consistent observation of radial inhomogeneity was made, which becomes most prominently visible in **Figure 6A**. More specifically, larger stresses and larger apparent deformations were observed at angles of 45°, 135°, 225°, and 315° with respect to the *x*- and *y*-axes than coincident with the axes. This inhomogeneity is the consequence of the discretization of a cylindrical object on a polar grid in the phantom bitmap image: Parallel to any axis, the phantom is exactly 10 pixels (2.3 x 10^{-4} m) thick, whereas rays at odd multiples of 45° detect the inner and outer boundaries approximately 8 pixels (1.84 x 10^{-4} m) apart. This apparent inhomogeneity is correctly recognized by the meshing algorithm and correctly visualized by the toolchain. However, it should be noted that this apparent inhomogeneity indicates the critical role of the object discretization and points at a potential source of error for all FEM toolchains when objects are extracted from medical images with low resolution.

In spite of its relative inflexibility, we believe that the meshing module is useful in its source form, because it demonstrates in detail how a Tochnog input file is composed. Therefore, the meshing module may readily be modified with other extraction algorithms (such as, e.g., the marching cubes algorithm) to accommodate a larger variety of shapes. For example, a simple extension towards spherical objects can be envisioned if the probing rays are emitted radially in all directions from the centroid. We did not further pursue this path, because the ray-based algorithm proved to be sufficient for our application, and because generalized algorithms based on the

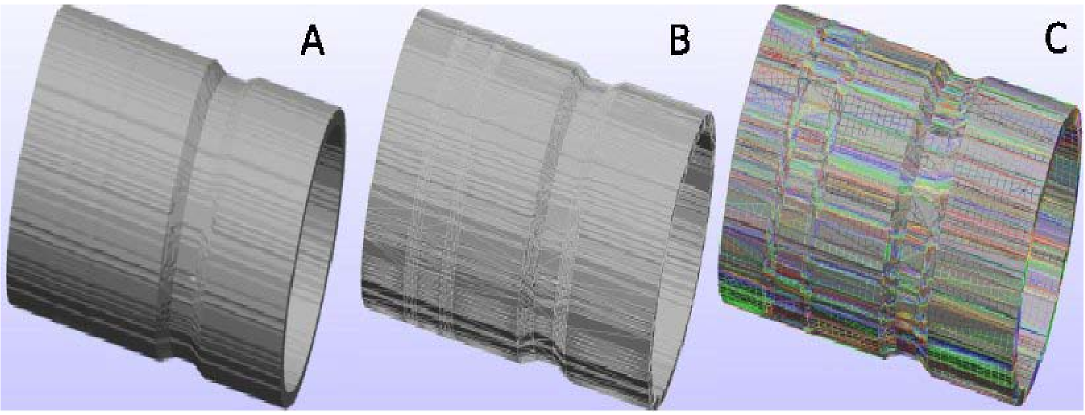


FIG. 9 VISUALIZATION RESULTS OF THE FIRST PHANTOM MODEL SUBJECTED TO COMMERCIAL IMAGE PRE-PROCESSING SOFTWARE THE GEOMETRIC RENDERING (A), THE MESHED MODEL (B), AND THE COLORED IDENTIFICATION OF ALL THE SURFACES AND ELEMENTS THAT THE MODEL IS COMPRISED OF (C) ARE HIGHLIGHTED.

marching-cubes algorithm exist [25-28]. By nature, familiarity with computer programming and the C/C++ language is helpful in understanding the algorithms and methods employed in the meshing module when a modification of our meshing algorithm is desired.

Visualization Software Options

Both of the open-source visualization software options evaluated are fully capable of providing suitable visualizations of FEM output data. Paraview is similarly intuitive as its commercial competitor, GiD. However, Paraview allowed us to obtain visualizations of the magnitude for stress, strain, and deformation values. This feature was not found in GiD; vector values are presented and must be calculated in order to obtain magnitude values. Conversely, Paraview does not allow us to to visualize the varying material groups along the inhomogeneous section containing lower image intensity values. In addition, we were unable to visualize the exaggerated expansion behavior of the vessel when subjected to internal pressure as presented in **Figure 6**. OpenDX is advantageous in the sense that it allowed us to superimpose multiple aspects of the simulation onto one image. For instance, in the visualization presented in **Figure 8**, we were able to view the volumetric rendering of the tube (in light gray), the stress magnitude (as points in space), the pressure distribution throughout the tube (diagonal circle at left end of tube), and the inhomogenous section of the tube containing lower image intensity values (black). Tochnog output files created for OpenDX are typically very large, because all data for each point in time is recorded for processing in OpenDX. To extract specific data from the Tochnog output file, for example, the shear stress magnitude, specific visual programs must be created. Unlike Paraview and GiD, which are very intuitive, creation of a visualization with OpenDX requires experience with OpenDX. The key advantage of OpenDX is the unmatched flexibility with which visualizations can be designed.

Conclusion

The open-source philosophy allows the presented FEM chain to be available for use, modifications, improvements, and free distribution by our biomedical research peers. Our proposed modeling chain exhibits versatility as it can be applied to any tubular biomedical object that has been subjected to biomedical imaging, including medical device tubing, stents, or long bones. The meshing module is our contribution to the open-source community, and it can be freely downloaded at http://haidekker.org/cimage/. In this study, our module was applied only to phantom models of tubular constructs, but we have provided the framework which allows us to subject tissue-engineered vascular grafts to FEM analysis to examine biomechanical behavior. Future research will revolve around continuing to apply our modeling chain to objects of varying geometrical shapes and types of materials to test the specificity and efficiency of the probing ray principle.

ACKNOWLEDGEMENTS

Funding for this study was in part provided by the National Institutes of Health, grant 1R21 HL081308.

We thank Dr. Nicolas L'Heureux (Cytograft Inc., Novato, CA) for providing us with the tissue-engineered blood vessel sample that is shown in **Figure 2**.

Appendix

Interface between Meshing Module and Tochnog

Summary of key elements in the Tochnog control file: any meshing program needs to produce a file that contains the components described in this section.

- The header section. The header contains general control elements, for example, the number of spatial dimensions, and the simulated values we want to observe. In our case, we chose to observe the velocity, stresses, the total and elastic strains of the 3-D tubular construct when subjected to homogeneous pressure.
- The nodes section. This section lists all vertices (nodes) and their spatial location. Therefore, this section describes the geometry of the object. Nodes must be shared between adjoining elements. An example of lines that describe a group of nodes follows:

node 0	4.000000	0.000000	0.000000
node 1	3.695518	1.530734	0.000000
node 2	5.000000	0.000000	0.000000
node 3	4.619398	1.913417	0.000000
node 4	4.000000	0.000000	1.000000
node 5	3.695518	1.530734	1.000000
node 6	5.000000	0.000000	1.000000
node 7	4.619398	1.913417	1.000000

- Nodes are numbered consecutively (the number after the keyword "node"), and the spatial coordinates in a Cartesian system follow. It can be seen in this example that the first four nodes belong to the lowest slice at *z*=0, and the next four nodes belong to the slice at *z*=1.
- The elements section. In this section, nodes are grouped to form a cuboid element. Several element geometries are available in the selected FEM software, but we made use of a cube, which is referred to as *hex8* element in the software's terminology. A sample element definition follows:

element 0 -hex8	0 1 2 3 4 5 6 7
element 1 -hex8	1 8 3 9 5 10 7 11
element 2 -hex8	8 12 9 13 10 14 11 15
element 3 -hex8	12 16 13 17 14 18 15 19

The number after the keyword "element" is the element number, used in material assignments. The instruction "-hex8" indicates the geometry, and the following eight integer numbers are the numbers of the nodes for this element. It can be seen that nodes are shared between adjoining elements. The order of the nodes is crucial. **Figure 5** shows the required order of the nodes.

- The element grouping section. In this section, each element is assigned to one material group. In one extreme case, only one material group exists, and each element is assigned to element group 0. In the other extreme case, variability is so high that each element has its own material group. Depending on material property binning, the number of material groups can be reduced substantially.
- The material properties section. In this section, material properties (among them, Young's modulus, Poisson elasticity, and material density) are defined. This is also the section where nonlinear material properties are introduced in the FEM software.
- The node boundary condition definitions. In this section nodes may be subjected to boundary constraints. In the special case that is considered in this study, all inner-wall nodes are subjected to a constant pressure, and all nodes in the lowest and uppermost slice are held fixed in space.
- The control section. In this section, the evolution in time of the simulation can be controlled. Factors include time steps, iteration limits, time intervals after which a snapshot is saved, and a control parameter of how the FEM software may subdivide the cubes if they become too inhomogeneous.

REFERENCES

[1] D. J. Hawkes, D.Barratt, J.M. Blackall, C. Chan, P.J. Edwards, K. Rhode, G.P. Penney, J.McClelland, and D.L.G. Hill, Medical Image Analysis 9, 2 (2005).

[2] A. Hagemann, K. Rohr, and H.S. Stiehl, Medical Image Analysis 6, 4 (2002).

[3] T.A. Sundaram and J.C. Gee, Medical Image Analysis 9, 6 (2005).

[4] J. Al-Sukhun, C. Lindqvist, N. Ashammakhi, and H. Penttilä, Br. J. Oral Maxillofac. Surg. 45, 2 (2007).

[5] C.J. Beller, M.R. Labrosse, M.J. Thubrikar, and F. Robicsek, Circulation 109, 6 (2004).

[6] C.J. Beller, M.R. Labrosse, M.J. Thubrikar, G.Szabo, F. Robicsek, and S.Hagl, Eur. J.Cardiothorac. Surg. 27, 2 (2005).

[7] D.N. Ghista, A.S. Kobayashi, and N.Davids, Computers in Biology and Medicine 5, (1975).

[8] G.H. Kwon, S.W. Chae, and K.J. Lee, Computers and Structures 81, (2003).

[9] Z. Yu, M.J. Holst, and J. A. McCammon, Finite Elements in Analysis and Design 44, 11 (2008).

[10] C.K. Chui, Z. Wang, J. Zhang, J.S.K. Ong, L. Bian, J.C.M. Teo, C.H. Yan, S.H. Ong, S.C. Wang, H.K. Wong, and S.H. Teoh, Advances in Engineering Software 40, 3 (2009).

[11] N.M. Grosland, K.H. Shivanna, V.A. Magnotta, N.A. Kallemeyn, N.A. DeVries, S.C. Tadepalli, and C. Lisle, Computer Methods and Programs in Biomedicine 94, 1 (2009).

[12] N. L'Heureux, S. Pâquet, R. Labbé, L. Germain, and F.A. Auger, The FASEB Journal 12, 1 (1998).

[13] N. L'Heureux, N. Dusserre, G. Konig, B. Victor, P. Keire, T.N. Wight, N. A. F. Chronos, A.E. Kyles, C.R. Gregory, and G. Hoyt, Nature medicine 12, 3 (2006).

[14] J.T. LaCroix, J. Xia, and M.A. Haidekker, Annals of Biomedical Engineering 37, 7 (2009).

[15] J.T. LaCroix, and M.A. Haidekker, BMC medical imaging 9, 1 (2009).

[16] H.M. Huang, J. Xia, and M.A. Haidekker, Annals of Biomedical Engineering 36, 10 (2008).

[17] C. Lally, F. Dolan, and P. J. Prendergast, Journal of Biomechanics 38, 8 (2005).

[18] P. J. Prendergast, C. Lally, S. Daly, A. J. Reid, T. C. Lee, D. Quinn, and F. Dolan, J. Biomech. Eng. 125, 5 (2003).

[19] W. Wu, W.Q. Wang, D.Z. Yang, and M. Qi, Journal of Biomechanics 40, 11 (2007).

[20] M. Haidekker, R. Andresen, C. Evertsz, D. Banzer, and H. Peitgen, The British Journal of Radiology 70, 834 (1997).

[21] L. Allard, G. Cloutier, and L. Durand, Acoustical Imaging 23, (1997).

[22] Z. Guo and A. Fenster, Ultrasound in Medicine and Biology 22, 8 (1996).

[23] Z. Guo, L. G. Durand, L. Allard, G. Cloutier, and A. Fenster, Journal of Vascular Surgery 27, 4 (1998).

[24] G. Cloutier, Z. Qin, D. Garcia, G. Soulez, V. Oliva, and L. G. Durand, Ultrasound in Medicine and Biology 26, 9 (2000).

[25] W. Lorensen, and H. Cline, Computer Graphics 21, 4 (1987).

[26] T. Elvins, ACM SIGGRAPH Computer Graphics 26, 3 (1992).

[27] C. Montani, R. Scateni, and R. Scopigno, IEEE Conference Visualization, 1994.

[28] T. Nishimara, and T. Fugimoto, Systems and Computers in Japan 25, 3 (1994).